

1 **CLAIMS**

2 I claim:

3 1. A method comprising organizing a byte stream of an information structure, said information
4 structure having a schema and an in-memory representation, said schema having a schema tree
5 representation with a plurality of schema nodes, said schema nodes including at least one leaf and
6 at least one interior node, the step of organizing comprising the steps of:

7 computing a layout from the schema tree representation by depth-first enumeration of leaf nodes
8 of the schema;

9 serializing the byte stream from the in-memory representation while grouping together all scalar
10 items from the in-memory representation corresponding to each schema node; and

11 accessing information from the byte stream by using the layout and offset calculations.

12 2. A method as recited in claim 1, wherein said information structure is a message.

13 3. A method as recited in claim 1, wherein the step of computing a layout comprises:

14 establishing a fixed length portion of the byte stream, the fixed length portion having a slot for
15 each enumerated schema leaf node; and

16 establishing a varying length portion of the byte stream following the fixed length portion, the
17 varying length portion having successive areas for any information items requiring varying length
18 encoding.

1 4. A method as recited in claim 1, wherein the step of computing a layout comprises:

2 establishing a fixed length portion of the byte stream, the fixed length portion having a slot for
3 each enumerated schema leaf node having a predecessor in the depth-first numbering requiring
4 varying length encoding; and

5 establishing a varying length portion of the byte stream following the fixed length portion, the
6 varying length portion having successive areas for each enumerated schema node.

7 5. A method as recited in claim 1, wherein interior nodes of said schema tree representation are
8 restricted to list and tuple nodes, and leaf nodes comprise scalar types and dynamic types.

9
10 6. A method as recited in claim 1, wherein the step of serializing the byte stream comprises:

11 determining a correspondence between the in-memory representation and the schema tree
12 representation;

13 initializing the byte stream by reserving a fixed length portion and pointing to a beginning of a
14 variable length portion;

15 retrieving a location in the byte stream for an element of the in-memory representation
16 information corresponding to a first schema leaf node in depth first order from the layout;

17 converting the element to bytes in the byte stream according to a number of elements
18 corresponding to the schema leaf node; and

19 repeating the steps of retrieving and converting for all schema leaf nodes in depth-first order.

1 7. A method as recited in claim 6, wherein the step of converting elements to bytes comprises
2 recording a nested list of tuples in column order rather than row order, resulting in a set of nested
3 lists.

4 8. A method as recited in claim 6, wherein the step of converting elements to bytes comprises
5 preceding each list of varying length items with an offset table allowing any element of said each
6 list to be reached in constant time from a head of said each list.

7 9. A method as recited in claim 1, wherein the step of accessing information comprises the steps
8 of:

9 scanning a list of key values representing a table column serialized within the byte stream to
10 determine an index position; and

11 using the index position in conjunction with offset calculations and offset tables serialized at the
12 start of lists within the byte stream to find information in lists representing non-key table
13 columns.

14 10. A method as recited in claim 1, wherein the schema tree representation is derived from a
15 schema graph representation by truncating recursive definitions and variants and replacing
16 truncated sub-trees with leaf nodes of a dynamic type.

17 11. A method as recited in claim 1, further comprising performing a preliminary reorganization
18 of the schema to distribute tuples over variants prior to carrying out the steps of computing,
19 serializing and accessing.

1 12. An article of manufacture comprising a computer usable medium having computer readable
2 program code means embodied therein for causing organization of a byte stream of an
3 information structure, the computer readable program code means in said article of manufacture
4 comprising computer readable program code means for causing a computer to effect the steps of
5 claim 1.

6 13. A program storage device readable by machine, tangibly embodying a program of
7 instructions executable by the machine to perform method steps for organizing a byte stream
8 form of an information structure, said method steps comprising the steps of claim 1.

9 14. An apparatus comprising a serializer/deserializer for a byte stream form of an information
10 structure, said information structure having a schema and an in-memory representation, said
11 schema having a schema tree representation with a plurality of schema nodes, said schema nodes
12 including at least one leaf and at least one interior node, the serializer/deserializer comprising:

13 a processor for computing a layout from the schema tree representation by depth-first
14 enumeration of leaf nodes of the schema;

15 a serializer for serializing the byte stream from the in-memory representation while grouping
16 together all scalar items from the in-memory representation corresponding to each schema node;
17 and

18 a selective de-serializer for accessing information from the byte stream by using the layout and
19 offset calculations.

1 15. An apparatus as recited in claim 14, wherein the processor comprises a module for
2 establishing a fixed length portion of the byte stream, the fixed length portion having a slot for
3 each enumerated schema leaf node; and for establishing a varying length portion of the byte
4 stream following the fixed length portion, the varying length portion having successive areas for
5 any information items requiring varying length encoding.

6 16. An apparatus as recited in claim 14, wherein the processor comprises a module for
7 establishing a fixed length portion of the byte stream, the fixed length portion having a slot for
8 each enumerated schema leaf node having a predecessor in the depth-first numbering requiring
9 varying length encoding; and for establishing a varying length portion of the byte stream
10 following the fixed length portion, the varying length portion having successive areas for each
11 enumerated schema node.

12 17. An apparatus as recited in claim 14, wherein the serializer comprises:

13 a reconciling module to determine a correspondence between the in-memory representation and
14 the schema tree representation;

15 an initialization module to initialize the byte stream by reserving a fixed length portion and
16 pointing to a beginning of a variable length portion;

17 a lookup module to retrieve a location in the byte stream for an element of the in-memory
18 representation information corresponding to a first schema leaf node in depth first order from the
19 layout;

20 a converter to convert the element to bytes in the byte stream according to a number of elements
21 corresponding to the schema leaf node, wherein all schema leaf nodes are retrieved and
22 converted in depth-first order.

1 18. An apparatus as recited in claim 17, wherein the converter comprises a recorder to record a
2 nested list of tuples in column order rather than row order, resulting in a set of nested lists.

3 19. An apparatus as recited in claim 17, wherein the converter precedes each list of varying
4 length items with an offset table allowing any element of said each list to be reached in constant
5 time from a head of said each list.

6 20. An apparatus as recited in claim 14, wherein the selective de-serializer scans a list of key
7 values representing a table column serialized within the byte stream to determine an index
8 position, and uses the index position in conjunction with offset calculations and offset tables
9 serialized at the starts of lists within the byte stream to find information in lists representing
10 non-key table columns.

11 21. An apparatus as recited in claim 14, wherein the schema tree representation is derived from a
12 schema graph representation by truncating recursive definitions and variants and replacing them
13 with leaf nodes of dynamic type.

14 22. An apparatus as recited in claim 14, wherein a preliminary reorganization of the schema is
15 performed to distribute tuples over variants prior to carrying out the remaining steps.

16 23. A computer program product comprising a computer usable medium having computer
17 readable program code means embodied therein for causing organization of a byte stream form of
18 an information structure, the computer readable program code means in said computer program
19 product comprising computer readable program code means for causing a computer to effect the
20 functions of claim 14.